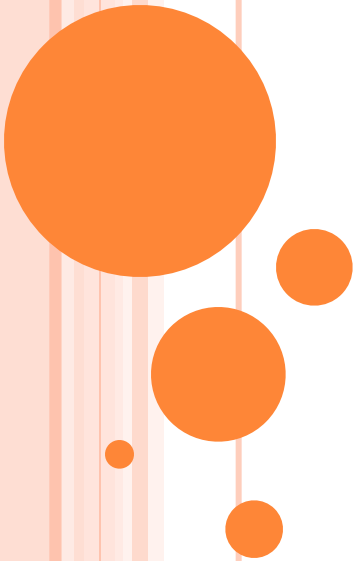
NODE.JS SERVER SIDE JAVASCRIPT



INTRODUCTION NODE.JS

Node.js was created by **Ryan Dahl** starting in 2009, and its growth is sponsored by Joyent, his employer.

WHAT IS NODE.JS ?

- Evented I/O for JavaScript
- Server Side JavaScript
- Runs on Google's V8 JavaScript Engine

WHY USE NODE.JS ?

- Node's goal is to provide an easy way to build scalable network programs.

WHAT IS UNIQUE ABOUT NODE.JS?

1. JavaScript used in client-side but node.js puts the JavaScript on server-side thus making communication between client and server will happen in same language

2. Servers are normally thread based but Node.JS is “Event” based. Node.JS serves each request in a Evented loop that can able to handle simultaneous requests.

WHAT IS UNIQUE ABOUT NODE.JS (CON'T)?

3. Node.JS programs are executed by V8 Javascript engine the same used by Google chrome browser.

HOW DOES IT DIFFER?

"Node is similar in design to and influenced by systems like Ruby's event machine or Python's twisted. Node takes the event model a bit further—it presents the event loop as a language construct instead of as a library."

WHAT CAN YOU DO WITH NODE ?

- It is a command line tool. You download a tarball, compile and install the source.
- It lets you Layered on top of the TCP library is a HTTP and HTTPS client/server.
- The JS executed by the V8 javascript engine (the ting that makes Google Chrome so fast)
- Node provides a JavaScript API to access the network and file system.

WHAT CAN'T DO WITH NODE?

- Node is a platform for writing JavaScript applications outside web browsers. This is not the JavaScript we are familiar with in web browsers. There is no DOM built into Node, nor any other browser capability.

- Node can't run on GUI, but run on terminal

THREADS VS EVENT-DRIVEN

Threads	Asynchronous Event-driven
Lock application / request with listener-workers threads	only one thread, which repeatedly fetches an event
Using incoming-request model	Using queue and then processes it
multithreaded server might block the request which might involve multiple events	manually saves state and then goes on to process the next event
Using context switching	no contention and no context switches
Using multithreading environments where listener and workers threads are used frequently to take an incoming-request lock	Using asynchronous I/O facilities (callbacks, not poll/select or O_NONBLOCK) environments

WHY NODE.JS USE EVENT-BASED?

In a normal process cycle the webserver while processing the request will have to wait for the IO operations and thus blocking the next request to be processed.

Node.JS process each request as events, The server doesn’t wait for the IO operation to complete while it can handle other request at the same time.

When the IO operation of first request is completed it will call-back the server to complete the request.

SYSTEM REQUIREMENTS

- Node runs best on the POSIX-like operating systems. These are the various UNIX derivatives (Solaris, and so on) or work a likes (Linux, Mac OS X, and so on).

- While Windows is not POSIX compatible, Node can be built on it either using POSIX compatibility environments (in Node 0.4x and earlier).

NODE.JS VS APACHE

1. It's fast
2. It can handle tons of concurrent requests
3. It's written in JavaScript (which means you can use the same code server side and client side)

Platform	Number of request per second
PHP (via Apache)	3187,27
Static (via Apache)	2966,51
Node.js	5569,30

CONCLUSION

- Node.js faster than apache but it more hungry system's CPU and memory

- Node.js use event based programming, it make the server doesn't wait for the IO operation to complete while it can handle other request at the same time